IN THE SPECIFICATION:

The Office Action objected to the algorithms cited on pages 36-39 under 37 C.F.R. 1.52 due to insufficient spacing between the lines.

Please replace Algorithm 1 beginning on page 37, line 16 with the following:

Algorithm 1 Simplified Random Line Fit

```
GenerateNormalizedLineParams(lparams, ptA,ptB)
{
 dX = ptA.x – ptB.x; dY = ptA.y-ptB.y; invNorm = 1.0/sqrt(dX*dX+dY*dY);
 lparams.a = dY * invNorm;
 lparams.b = -dX*invNorm;
 lparams.c = -(lparams.a*ptA.x+lparams.b*ptA.y);
}

SimpleRandomFit(P, fitLineParameters, distanceThs, strongLineRatio)
{
 stop = false;
 numMaxRejectPts = (1-strongLineRatio)*size(P);
 trials=0;
 while (not(stop) AND (trials < maxTrials)) {
        GenerateRandomPair(p,q);
        //Grab points from set P
        ptA= P(p); ptB= P(q);
        lparams = GenerateNormalizedLineParams(ptA,ptB);
        numRejectPts=0; ptIndex=0; setToStart(storePtIndex );
        while ((ptIndex < (size(P)-1)) AND (numRejectPts<numMaxRejectPts)) {
                if (DistanceLinetoPt(lparams,P(ptIndex)) > distanceThs)
                        numRejectPts++;
                else
```

```
                    ++storePtIndex = ptIndex;
                increment(ptIndex);
            }
            stop = numRejectPts<numMaxRejectPts;
            trials++;
        }
        fitLineParameters = FitLineTo(storePtIndex, P);
    }
```

Please replace Algorithm 2 beginning on page 40, line 1 with the following:

Algorithm 2 Random Line Fit

```
RandomLineFit(P, fitLineParameters, distanceThs, strongLineRatio,subsetThsRatio,
                        firstPassSizeRatio) {
    numberOfPoints = size(P);
    subsetSize = round(firstPassSizeRatio*numberOfPoints);
    numberOfValidPoints = 0;
    trials = 0;
    RandomizePoints(points, numberOfPoints);
    do {
        GenerateRandomLine(P, numberOfPoints, &lineParams);
        Increment(trials);
        //Check a subset of the total set
        numberOfValidPoints = CheckSubsetPoints(lineParams, P, distanceThs,
                                                    subsetSize,indiceOfValidPoints);
        if (numberOfValidPoints >= subsetThsRatio*subsetSize) {
            numberOfValidPoints+=CheckRemainingPoints(lineParams, P
                                                    distanceThs,indiceOfValidPoints);
            //If a better solution was found
            if (bestFoundNumberAccepted < numberOfValidPoints) {
```

```
            bestFoundNumberAccepted = numberOfValidPoints;
            bestLineParams = lineParams;
            bestIndiceValidPoints = indiceOfValidPoints;
        }
      }
  } while ((numberOfValidPoints < StrongLineRatio*numberOfPoints) AND
                                                    (trials< Max_Trials));
  //if desired a MSE fit can be done, but is not required.
  FitSetPointsMSE(points, bestIndiceValidPoints,&lineParams)
}
```

The Office Action requested a correction to the Specification beginning on page 1, last line regarding a reference to "Figures 6A-C" due to there being no Figures 6A-C in the Application. Applicants request the following amendments to correct this error:

Please replace the paragraph beginning on page 1, line 25, which begins, "If the outliers are few..." with the following paragraph:

If the outliers are few and not very far from the ideal line, standard statistical robust techniques can be used to find the best line fitting the set. For more extreme situations more complicated methods exist but tend to be computationally very complex. In some applications, such as image processing, the outliers can be numerous and wide-ranging. In certain problems several lines may be present in a set (see Figures 4 and 610A-C). Usually one is interested in finding the strongest line in the set, which is the line defined by the largest number of points.